



# VHDL

## 实例剖析

张凯 林伟 编著



国防工业出版社

National Defense Industry press <http://www.ndip.cn>

# VHDL 实例剖析

张凯 林伟 编著

国防工业出版社

·北京·

## 内 容 提 要

本书内容新颖、以实例见长,对涉及到 VHDL 语言的语法和工具都用实例来加以诠释。

本书分为三个部分对 VHDL 语言进行全面介绍。第一部分为基础篇,介绍了 VHDL 语言的基础知识,主要目的是为初学者打下一个良好的基础;第二部分是提高篇,介绍了 VHDL 语言的高级知识和目前在电子电路设计中常用电路结构的 VHDL 描述,目的是使读者掌握用 VHDL 语言设计简单逻辑电路的基本方法;第三部分是应用篇,从一些大型实例出发,介绍用 VHDL 语言设计大型复杂电路的流程和在设计过程中所用到的设计技巧,并且简要介绍了可编程逻辑器件的基础知识和 XILINX 系统软件的基本使用。书中列举的大量实例都经过微机上的 XILINX 软件或 HP 工作站上的 Synopsys 软件的验证。

本书的特点是技术实用、易读易懂、由浅入深,能够使读者逐步掌握 VHDL 语言。本书不仅注重基础知识的介绍,而且力求向读者系统地讲解 VHDL 的实际应用。它既可作为高等学校计算机和电子工程专业的研究生、本科生的教材和教学参考书,也可作为广大电子电路设计工程师、ASIC 设计人员和系统设计人员的参考书。

### 图书在版编目(CIP)数据

VHDL 实例剖析/张凯,林伟编著. —北京:国防工业出版社,2004.1

ISBN 7-118-03278-6

I.V... II.①张...②林... III. 硬件描述语言, VHDL—程序设计 IV.TP312

中国版本图书馆 CIP 数据核字(2003)第 088833 号

国防工业出版社出版发行

(北京市海淀区紫竹院南路 23 号)

(邮政编码 100044)

北京奥隆印刷厂印刷

新华书店经售

\*

开本 787×1092 1/16 印张 29¼ 682 千字

2004 年 1 月第 1 版 2004 年 1 月北京第 1 次印刷

印数:1—3000 册 定价:39.00 元

---

(本书如有印装错误,我社负责调换)

# 前 言

通信技术带动了整个电子工业的发展,可编程逻辑器件的应用极为广泛。ASIC 的开发和板级控制的应用使集成电路的设计朝着速度快、性能高、容量大、体积小和微功耗的方向发展,这种发展必将导致 VHDL 的工程设计规模日益增大,复杂程度日益增高。在这种情况下,单纯的语法介绍已经不能满足 EDA 设计者的需要,所以本书应运而生。

由于设计的复杂性,必须有一种语言能在各个层面上精确地为各种电路行为和结构建立模型,以便在计算机上对设计是否正确进行仿真。VHDL 受到在第一线工作的设计工程师的特别青睐,不仅因为 VHDL 与 C 语言很相似,学习和掌握起来并不困难,更重要的是它在复杂的 SOC 的设计上所显示的非凡性能和可扩展能力。

在数字系统设计的仿真领域,VHDL 早在 10 多年前就已得到全世界数字系统设计工程师的广泛承认,是目前世界上最普及的硬件描述语言。近年来它在数字系统自动综合方面也显示出旺盛的生命力。

VHDL 语言是一种用于电路设计的高级语言,出现于 20 世纪 80 年代的后期。最初是由美国国防部开发出来供美军提高设计的可靠性和缩减开发周期的一种使用范围较小的设计语言。但是,由于它在一定程度上满足了当时的设计需求,所以于 1987 年成为 ANSI/IEEE 的标准 (IEEE STD 1076-1987),1993 年更进一步修订,变得更加完备,成为 ANSI/IEEE 的 ANSI/IEEE STD 1076-1993 标准。目前,大多数的 CAD 厂商出品的 EDA 软件都兼容了这种标准。

随着 VHDL 语言的广泛应用,无论是电子设计工程师,还是高等院校的学生都迫切需要一本除了介绍 VHDL 基本概念和基本语法外,还能够从实际出发、着重介绍各种不同电路结构的描述方法的参考书。作者编写此书的目的就是从以上方面来满足广大读者的需要,以使读者能够全面掌握 VHDL 语言。除此之外本书提供了一些大型 VHDL 设计的例子,让大家了解开发一个大型 VHDL 项目的基本思路和方法,最后我们作了有益的补充,从 VHDL 的编程风格上让大家的编程水平和思路有一个较大的提高。

本书分为三个部分对 VHDL 语言进行全面介绍。

第一部分为基础篇,主要介绍 VHDL 语言的基础知识,目的是方便大家对 VHDL 的基本语法的查询和理解。该部分由第 1 章到第 6 章组成,重点介绍了 VHDL 概述、VHDL 的基本结构、VHDL 的基本语法、VHDL 语言程序的基本描述语句、VHDL 语言程序的高级特征;第 6 章为小型的实例介绍。

第二部分为提高篇,介绍了业界常用的 VHDL 工具 XILINX 开发的 ISE4.2i 和 ModelSim,采用由浅入深的介绍方法帮助大家逐步掌握 XILINX 工具。本章的介绍是通过实例来进行的,为了大家能够清楚地了解 XILINX 工具的使用流程,作者提供了大量插图,读者能够按图索骥。章节的划分为从第 7 章到第 10 章。

第三部分是应用篇，从一些大型实例出发，目的是使读者除了掌握用 VHDL 语言设计简单逻辑电路的基本方法之外，还能对实际的 VHDL 工程应用有所了解。在大型实例部分需要大家投入足够的精力，因为这一部分是前面所有章节的集合，它折射出大型项目的设计方法和在不同情况下 VHDL 语言的使用。本篇包括第 11 章和第 12 章：着重介绍了 VHDL 在数据通信领域常用的实例和 VHDL 编程风格。

本书中列举的实例取自于实际硬件电路的设计，并且经过了 PC 上的 XINLINX 软件或 SUN SPARC 工作站上的 Synopsys 软件的综合验证。实例源码可以从 <http://www.ndip.cn/wwwroot/compwter/index.htm> 下载使用。

本书的特点是实例优先、语法兼重、由浅入深，能够使读者逐步掌握 VHDL 语言并通过实例系统地了解 VHDL 的实际应用。它既可作为高等学校计算机、通信和电子工程专业的研究生、本科生的教材和教学参考书，也可作为广大电子电路设计工程师、ASIC 设计人员和系统设计人员的参考书。

本书是作者多年来使用 VHDL 语言设计硬件电路的经验总结，同时在写作过程中参考了大量有关 VHDL 语言的最新专著及文献。李敏、陈勇、张逸凡参与了本书部分章节的编写和校对工作。在此，向所有对本书作出贡献的同志表示衷心的感谢！

由于作者的水平有限，书中难免存在错误和不足之处，望读者给予批评和指正。

作 者

# 目 录

## 第一部分 基础篇

|                         |    |
|-------------------------|----|
| 第 1 章 概述                | 1  |
| 1.1 硬件描述语言              | 1  |
| 1.1.1 硬件描述语言的产生         | 1  |
| 1.1.2 利用硬件描述语言设计硬件电路的方法 | 2  |
| 1.1.3 硬件描述语言的种类         | 3  |
| 1.2 VHDL 硬件描述语言         | 4  |
| 1.2.1 VHDL 语言的历史        | 4  |
| 1.2.2 VHDL 语言的主要优势      | 4  |
| 1.2.3 VHDL 语言的不足之处      | 6  |
| 1.2.4 VHDL 语言的设计流程      | 6  |
| 第 2 章 VHDL 语言程序的基本模型结构  | 7  |
| 2.1 VHDL 语言程序的基本结构      | 7  |
| 2.1.1 实体说明              | 8  |
| 2.1.2 结构体               | 12 |
| 2.2 VHDL 语言结构体的三种描述方法   | 14 |
| 2.2.1 结构体的行为描述          | 14 |
| 2.2.2 结构体的数据流描述         | 16 |
| 2.2.3 结构体的结构化描述         | 17 |
| 2.3 VHDL 语言结构体的子结构描述    | 19 |
| 2.3.1 块语句结构             | 19 |
| 2.3.2 进程语句结构            | 21 |
| 2.3.3 子程序语句结构           | 24 |
| 第 3 章 VHDL 语言程序的基本语法    | 32 |
| 3.1 数据对象                | 32 |
| 3.1.1 常量                | 32 |
| 3.1.2 变量                | 33 |
| 3.1.3 信号                | 35 |
| 3.1.4 文件                | 36 |
| 3.2 数据类型                | 37 |
| 3.2.1 标准的数据类型           | 37 |

|              |                         |           |
|--------------|-------------------------|-----------|
| 3.2.2        | 用户定义的数据类型               | 39        |
| 3.2.3        | 用户定义的子类型数据              | 43        |
| 3.2.4        | 数据类型的转换                 | 43        |
| 3.3          | 操作运算符                   | 44        |
| 3.3.1        | 逻辑运算符                   | 45        |
| 3.3.2        | 算术运算符                   | 46        |
| 3.3.3        | 关系运算符                   | 47        |
| 3.3.4        | 并置运算符                   | 47        |
| 3.3.5        | 运算符的优先级                 | 48        |
| 3.4          | 词法描述                    | 49        |
| 3.4.1        | 字符集                     | 49        |
| 3.4.2        | 词法单元                    | 50        |
| 3.4.3        | 分界符                     | 50        |
| 3.4.4        | 标识符                     | 51        |
| 3.4.5        | 注释                      | 52        |
| 3.4.6        | 字符和字符串文字                | 53        |
| 3.4.7        | 位串文字                    | 53        |
| 3.4.8        | 抽象文字                    | 54        |
| <b>第 4 章</b> | <b>VHDL 语言程序的基本描述语句</b> | <b>55</b> |
| 4.1          | 顺序语句                    | 55        |
| 4.1.1        | 顺序赋值语句                  | 55        |
| 4.1.2        | WAIT 语句                 | 58        |
| 4.1.3        | IF 语句                   | 61        |
| 4.1.4        | CASE 语句                 | 66        |
| 4.1.5        | LOOP 语句                 | 68        |
| 4.1.6        | NEXT 语句                 | 70        |
| 4.1.7        | EXIT 语句                 | 71        |
| 4.1.8        | RETURN 语句               | 72        |
| 4.1.9        | NULL 语句                 | 72        |
| 4.1.10       | 断言语句                    | 73        |
| 4.1.11       | REPORT 语句               | 74        |
| 4.2          | 并行语句                    | 74        |
| 4.2.1        | 并行信号赋值语句                | 75        |
| 4.2.2        | 块语句                     | 81        |
| 4.2.3        | 进程语句                    | 83        |
| 4.2.4        | 过程调用语句                  | 85        |
| 4.2.5        | 参数传递语句                  | 86        |
| 4.2.6        | 元件例化语句                  | 88        |
| 4.2.7        | 生成语句                    | 90        |

|                                    |     |
|------------------------------------|-----|
| 4.2.8 并行断言语句·····                  | 94  |
| <b>第5章 VHDL 语言程序的高级特征</b> ·····    | 96  |
| 5.1 库·····                         | 96  |
| 5.1.1 库的基本概念·····                  | 96  |
| 5.1.2 常见的库·····                    | 97  |
| 5.2 程序包·····                       | 98  |
| 5.2.1 程序包的基本概念·····                | 98  |
| 5.2.2 常见的程序包·····                  | 102 |
| 5.3 配置·····                        | 103 |
| 5.3.1 默认连接和默认配置·····               | 103 |
| 5.3.2 元件的配置·····                   | 104 |
| 5.3.3 块的配置·····                    | 106 |
| 5.3.4 结构体的配置·····                  | 107 |
| 5.4 VHDL 语言中的属性描述·····             | 108 |
| 5.4.1 数值属性·····                    | 109 |
| 5.4.2 函数属性·····                    | 113 |
| 5.4.3 信号属性·····                    | 119 |
| 5.4.4 数据类型属性·····                  | 124 |
| 5.4.5 数据范围属性·····                  | 124 |
| 5.5 重载·····                        | 125 |
| 5.5.1 子程序重载·····                   | 125 |
| 5.5.2 运算符重载·····                   | 134 |
| <b>第6章 常用逻辑电路的 VHDL 语言程序</b> ····· | 136 |
| 6.1 常用组合逻辑电路的 VHDL 语言程序·····       | 136 |
| 6.1.1 门电路·····                     | 136 |
| 6.1.2 编码器·····                     | 141 |
| 6.1.3 译码器·····                     | 145 |
| 6.1.4 选择器·····                     | 150 |
| 6.1.5 缓冲器·····                     | 154 |
| 6.1.6 运算器·····                     | 156 |
| 6.2 常用时序逻辑电路的 VHDL 语言程序·····       | 158 |
| 6.2.1 触发器·····                     | 158 |
| 6.2.2 寄存器·····                     | 165 |
| 6.2.3 移位寄存器·····                   | 167 |
| 6.2.4 计数器·····                     | 172 |

## 第二部分 提高篇

|                          |     |
|--------------------------|-----|
| <b>第7章 小型的设计实例</b> ····· | 178 |
| 7.1 七段字符显示器·····         | 178 |



|              |                                    |            |
|--------------|------------------------------------|------------|
| 7.2          | 加法器 .....                          | 183        |
| 7.3          | 分频电路 .....                         | 187        |
| 7.4          | 8×9 FIFO 电路 .....                  | 191        |
| 7.5          | 内存控制器 .....                        | 195        |
| <b>第 8 章</b> | <b>XILINX ISE4.2i 快速入门 .....</b>   | <b>199</b> |
| 8.1          | XILINX ISE4.2i 概述及设计流程 .....       | 199        |
| 8.2          | 设计开始 .....                         | 200        |
| 8.2.1        | 软件的安装 .....                        | 200        |
| 8.2.2        | 运行 ISE 软件 .....                    | 203        |
| 8.2.3        | 使用在线帮助 .....                       | 209        |
| 8.3          | 设计输入 VHDL .....                    | 209        |
| 8.3.1        | 创建一个新的工程项 .....                    | 209        |
| 8.3.2        | 创建一个计数器模块 .....                    | 210        |
| 8.3.3        | 利用计数器模板修改计数器模块 .....               | 211        |
| 8.4          | 仿真行为模型功能仿真 .....                   | 212        |
| 8.4.1        | 创建一个 testbench 波形源文件 .....         | 212        |
| 8.4.2        | 初始化计数器输入 .....                     | 213        |
| 8.4.3        | 生成预期的输出响应 .....                    | 214        |
| 8.5          | 使用 ModelSim 进行仿真 .....             | 214        |
| 8.5.1        | 行为仿真 .....                         | 214        |
| 8.5.2        | 布局布线后的仿真 .....                     | 215        |
| <b>第 9 章</b> | <b>XILINX ISE 4.2i 的高级应用 .....</b> | <b>217</b> |
| 9.1          | 设计输入(Schematic 原理图输入) .....        | 217        |
| 9.1.1        | 创建 VHDL 模块生成一个原理图符号 .....          | 217        |
| 9.1.2        | 创建一个新的顶层原理图 .....                  | 217        |
| 9.1.3        | 例化 VHDL 模块 .....                   | 218        |
| 9.1.4        | 原理图中连线 .....                       | 218        |
| 9.1.5        | 为连线添加网络名 .....                     | 219        |
| 9.1.6        | 为总线添加网络名 .....                     | 220        |
| 9.1.7        | 添加输入/输出管脚标记 .....                  | 221        |
| 9.2          | 设计输入(FSM 状态机输入) .....              | 221        |
| 9.2.1        | 状态机的建立生成 VHDL 代码 .....             | 222        |
| 9.2.2        | 设计的功能仿真并产生 TestBench .....         | 228        |
| 9.3          | 设计输入(HDL 硬件描述语言输入方式) .....         | 229        |
| 9.3.1        | 层次性结构的描述 .....                     | 230        |
| 9.3.2        | 上层文件和下层文件的连接方式 .....               | 232        |
| 9.3.3        | 一个包含底层文件的实例 .....                  | 235        |
| 9.4          | 设计实现 .....                         | 251        |
| 9.4.1        | 运行实现设计 .....                       | 252        |

|                                          |            |
|------------------------------------------|------------|
| 9.4.2 在资源分配器 Floorplanner 中查看设计布局 .....  | 252        |
| 9.5 对顶层文件进行时序仿真 .....                    | 253        |
| 9.5.1 创建一个 testbench 波形源文件 .....         | 253        |
| 9.5.2 初始化计数器输入 .....                     | 254        |
| 9.5.3 生成预期的输出响应 .....                    | 254        |
| 9.5.4 布局布线后的仿真 .....                     | 255        |
| 9.6 IP 包的嵌入使用 .....                      | 256        |
| <b>第 10 章 CPLD/FPGA 的边界扫描与下载方式 .....</b> | <b>262</b> |
| 10.1 引言 .....                            | 262        |
| 10.2 边界扫描测试的结构 .....                     | 263        |
| 10.3 测试逻辑的控制 .....                       | 267        |
| 10.4 边界扫描的具体应用 .....                     | 270        |
| 10.5 XILINX 器件的下载 .....                  | 274        |
| 10.5.1 XILINX 器件的下载电缆 .....              | 274        |
| 10.5.2 XILINX 器件的下载方式 .....              | 278        |

### 第三部分 应用篇

|                                                |            |
|------------------------------------------------|------------|
| <b>第 11 章 VHDL 在数据通信领域的实际应用 .....</b>          | <b>282</b> |
| 11.1 UTOPIA2 接口实例 .....                        | 282        |
| 11.1.1 UTOPIA2 接口的原理描述 .....                   | 282        |
| 11.1.2 VHDL 程序 .....                           | 286        |
| 11.2 UART 的实例设计 .....                          | 304        |
| 11.2.1 UART 的结构 .....                          | 304        |
| 11.2.2 UART 的内部寄存器 .....                       | 305        |
| 11.2.3 UART 的发送器 .....                         | 316        |
| 11.2.4 UART 的接收器 .....                         | 325        |
| 11.2.5 UART 的中断状态机和管脚说明 .....                  | 335        |
| 11.2.6 MODEM 控制模块 .....                        | 347        |
| 11.3 使用 EPLD 配置 FPGA 的实例 .....                 | 349        |
| 11.3.1 使用 EPLD 配置 FPGA 的原理 .....               | 349        |
| 11.3.2 VHDL 程序 .....                           | 350        |
| 11.4 高速 FIFO 的实例 .....                         | 355        |
| 11.4.1 高速 FIFO 的原理 .....                       | 355        |
| 11.4.2 同步 FIFO 的 VHDL 程序 .....                 | 358        |
| 11.4.3 异步 FIFO 的 VHDL 程序 .....                 | 363        |
| 11.5 使用 FPGA 内部的 DLL .....                     | 374        |
| 11.5.1 FPGA 中 DLL 的功能介绍 .....                  | 374        |
| 11.5.2 VHDL 程序 1-DLL 的标准使用 .....               | 379        |
| 11.5.3 VHDL 程序 2-DLL 的为其他非 VIRTEX 芯片提供时钟 ..... | 384        |

|                                    |            |
|------------------------------------|------------|
| 11.6 可综合 200MHz 的 ZBTSRAM 接口 ..... | 386        |
| 11.6.1 接口原理描述 .....                | 386        |
| 11.6.2 VHDL 程序 .....               | 388        |
| <b>第 12 章 VHDL 编程风格与编码指南 .....</b> | <b>401</b> |
| 12.1 说明 .....                      | 401        |
| 12.2 VHDL 编码风格 .....               | 401        |
| 12.2.1 文件头 .....                   | 401        |
| 12.2.2 联机注释 .....                  | 403        |
| 12.2.3 命名规则 .....                  | 403        |
| 12.3 VHDL 编码指导 .....               | 404        |
| 12.3.1 在 VHDL 编码中层次化设计 .....       | 404        |
| 12.3.2 关于锁存和寄存器 .....              | 405        |
| 12.3.3 关于元件例化和黑箱操作 .....           | 408        |
| 12.3.4 DATAPATH 的分析 .....          | 408        |
| <b>附录 A VHDL 的保留字 .....</b>        | <b>415</b> |
| <b>附录 B VHDL 的标准包集合文件 .....</b>    | <b>417</b> |
| <b>参考文献 .....</b>                  | <b>460</b> |

# 第一部分 基础篇

## 第1章 概述

### 1.1 硬件描述语言

本小节将重点介绍硬件描述语言的基本知识，这些基本知识主要包括硬件描述语言的产生、硬件描述语言设计电路的方法以及常见硬件描述语言的种类。

#### 1.1.1 硬件描述语言的产生

熟悉硬件电路的开发人员都知道，传统的硬件电路设计方法主要是采用电路原理图的设计方法。这种方法最终所形成的设计文件是若干张电路原理图，在这些电路原理图中详细地标明了构成硬件电路的各种逻辑器件的名称和各个器件之间的信号连线关系。对于小规模硬件电路设计来说，采用这种设计方法只需要几十张至几百张电路原理图就可以完成；而对于大规模的硬件电路设计来说，采用这种设计方法往往需要几千张甚至几万张电路原理图。如此之多的电路原理图，往往会给该硬件电路设计的归档、阅读、修改和使用带来极大的不便。虽然说传统的硬件电路设计方法已经沿用了几十年，是广大设计工程师所熟悉和掌握的一种方法，但是随着科学技术和大规模集成电路的发展，这种传统的设计方法已经大大落后于当今技术的迅猛发展。因此，一种崭新的硬件电路设计方法随之迅速兴起，这种设计方法就是采用硬件描述语言的硬件电路设计方法，它的出现给硬件电路设计带来了一次重大的变革。

硬件描述语言（HDL，Hardware Description Language）发展至今已经有了 40 多年的历史，现在硬件描述语言已经成功地应用于硬件电路设计的模拟验证和综合优化等方面。所谓硬件描述语言，顾名思义，就是指对硬件电路系统进行行为描述、数据流描述或者结构化描述的一种语言。这种硬件描述语言的最大特点是可以借鉴高级编程语言的功能特性对硬件电路的行为和结构进行高度抽象化和规范化的描述；同时，还可以对硬件电路的设计进行不同层次、不同领域的模拟验证和综合优化等处理，从而使硬件电路的设计达到高度自动化。

一般来说，硬件描述语言主要包括两大类：一种是文字硬件描述语言，另一种是图形硬件描述语言。在实际硬件电路的设计过程中，通常这两种硬件描述语言可以并用。其中，文字硬件描述语言常常应用于数字硬件电路系统的设计，它标志着现代硬件电路设计方法的产生、成熟和发展；而图形硬件描述语言就是采用直观的图形来描述硬件电

路系统,例如逻辑电路图和状态流程图等,由于它与传统的硬件电路设计方法类似,因此将其称为硬件电路设计的古典方法。实际上,这两种硬件描述语言都十分重要,各有其优缺点,一个优秀的硬件电路设计师掌握这两种硬件描述语言是十分必要的。

现在,国内外硬件描述语言的种类十分繁多,有一些硬件描述语言是由 C 语言发展而来的,也有一些是由 PASCAL 语言发展而来的。其中,有些硬件描述语言已经成为 IEEE 标准,但是大多数的硬件描述语言都只是一种企业标准。目前我国广泛使用的硬件描述语言主要有三种类型:AHDL 语言、Verilog HDL 语言和 VHDL 语言。其中,AHDL 语言由于 ALTERA 公司的竭力推广,使得该语言在国内各大公司和各个大学有着十分广泛的用户群体,因此使用 AHDL 语言进行硬件电路设计的人员比例很高。但是,随着时间的推移,特别是 Verilog HDL 语言和 VHDL 语言成为 IEEE 标准后,国内各大公司和各个大学的设计人员已经将眼光转到了这两种硬件描述语言上来。近十年来,Verilog HDL 语言和 VHDL 语言在我国得到了迅速发展,它的用户群体与日俱增。

### 1.1.2 利用硬件描述语言设计硬件电路的方法

归纳起来,利用硬件描述语言设计硬件电路的方法主要有以下几个特点。

#### 1) 采用自顶向下的设计方法

一般来说,传统的硬件电路设计方法主要采用自底向上的设计方法。这种设计方法的主要原理是:首先定义子模块,进行子模块的具体设计;然后再根据设计的总体要求,将各个功能子模块合成,以完成硬件的总体设计。而采用硬件描述语言设计硬件电路常常采用自顶向下的设计方法,这种设计方法的主要原理是:首先确定顶层模块,进行顶层模块的设计,然后将顶层模块中的逻辑功能划分为不同的功能子模块,最后再进行功能子模块的详细设计。

通过这两种设计方法的原理不难看出,与传统的自底向上的设计方法相比,自顶向下的设计方法有助于在设计的早期发现结构设计中的错误,提高设计的一次成功率。

#### 2) 硬件电路系统中大量采用 ASIC 芯片

目前在硬件电路系统的设计中大量采用了 ASIC 芯片,一是由于 ASIC 芯片具有许多其他芯片所不具备的特点;二是众多的 ASIC 芯片厂商的工具软件都可以支持硬件描述语言,从而给设计带来极大的方便。这样,设计人员在设计硬件电路系统的过程中,就不会受到只能使用通用元器件的限制,而是可以根据硬件电路系统设计的需要来设计自用的 ASIC 芯片或可编程逻辑器件,从而使硬件电路系统的设计更加合理,使体积、重量和功耗等参数大大减小。

#### 3) 降低了硬件电路系统的设计难度

硬件描述语言具有多层次描述系统硬件功能的能力,可以从系统的数学模型直到门级电路。另外,硬件描述语言高层次的行为描述可以与低层次的寄存器传输描述和结构描述混合使用,从而使硬件电路系统的描述更加准确。同时,某些硬件描述语言还具有子程序调用功能,对于已经完成的设计程序,我们可以通过修改子程序的方法来改变设计的规模和结构,使得对硬件电路系统的描述更加灵活。上述这些特点,在某种程度上大大提高了设计人员的工作效率,降低了硬件电路系统的设计难度。

#### 4) 可以使设计的硬件电路系统迅速上市,降低成本

硬件描述语言与可编程逻辑很好地结合,大大提高硬件电路系统的实现速度。采用硬件描述语言设计硬件电路系统可以使设计更加方便、快捷,而可编程逻辑的应用可以将产品的前期风险降至最低,并使设计的快速复制简单易行,同时多种综合工具支持这种形式的设计。硬件描述语言和可编程逻辑的组合作为一种强有力的设计方式,可以使设计的硬件电路系统迅速上市,降低成本。

#### 5) 设计文件是采用硬件描述语言编写的程序

前面已经提到过,传统的硬件电路设计方法主要是采用电路原理图的设计方法,这种设计方法最终所形成的设计文件是若干张电路原理图;而采用硬件描述语言设计硬件电路系统时,最终形成的设计文件是采用硬件描述语言编写的程序,而且它可以转换成电路原理图的形式输出。采用硬件描述语言编写的程序作为设计文件可以使设计文件的资料量小、易于保存、便于阅读和修改方便。

### 1.1.3 硬件描述语言的种类

现在,国内外硬件描述语言的种类十分繁多,甚至各个企业和科研单位都有自己的硬件描述语言,下面将对一些常用的硬件描述语言进行简单介绍。

#### 1) ABEL 语言

ABEL 硬件描述语言(简称为 ABEL 语言)是美国 DATAIO 公司开发的高级可编程逻辑设计语言。ABEL 语言支持用各种类型的可编程逻辑器件来实现数字硬件电路系统的逻辑设计,它是目前流行的硬件描述语言之一。

ABEL 语言的特点主要包括以下几点:

- ① 支持所有可编程逻辑器件来实现逻辑设计。
- ② 用户设计文件结构化,便于编写和阅读。
- ③ 逻辑描述方式和设计方法灵活。
- ④ 一般设有程序模拟和测试用的测试向量。
- ⑤ ABEL 语言软件处理程序齐全、功能强大。
- ⑥ 对于计算机等硬件系统适应性强,硬件支持要求较低。

#### 2) AHDL 语言

AHDL 语言是由 ALTERA 公司发明的一种硬件描述语言。20 世纪 90 年代,由于 ALTERA 公司的竭力推广,使得 AHDL 语言在国内各大公司和各个大学有着十分广泛的用户群体,因此使用 AHDL 语言进行硬件电路设计的人员比例很高。AHDL 语言的特点是易学易用,有过高级编程语言基础的读者可以在很短的时间内掌握这门语言。它的缺点是移植性较差,一般只能用在 ALTERA 公司自己的开发系统。

#### 3) Verilog HDL 语言

Verilog HDL 语言是在使用最广泛的 C 语言的基础上发展起来的一种常用的硬件描述语言,它是由 GDA 公司的 PhilMoorby 在 1983 年发明的。最初,PhilMoorby 只是设计了一个仿真和验证的工具,之后又陆续开发了相关的故障模拟和时序分析的一系列工具。1985 年,PhilMoorby 推出了它的第三个商用仿真器 Verilog-XL,从而使 Verilog HDL 语言得到了迅速发展和推广应用。1989 年,CADENCE 公司收购了 GDA 公司,使 Verilog HDL 语言成为该公司的独家专利。1990 年,CADENCE 公司公开发表了 Verilog HDL 语言,

并且成立 LVI 组织以促使 Verilog HDL 语言成为 IEEE 标准,即 IEEE Standard 1364-1995。

Verilog HDL 语言的最大特点是易学易用,如果初学者有 C 语言的基础,那么他就可以在很短的时间内掌握这门语言。它的缺点是有着非常自由的语法,因此初学者容易犯一些设计上的错误。

#### 4) VHDL 语言

有关 VHDL 语言将在下一节中进行详细地介绍,这里就不另行介绍了。

## 1.2 VHDL 硬件描述语言

目前,VHDL 硬件描述语言已经成为各家 EDA 工具厂商和集成电路厂商所普遍认同和共同推广的标准化硬件描述语言。因此,掌握采用 VHDL 硬件描述语言设计硬件电路系统的相关知识,是每一个从事硬件电路设计的设计人员所必须掌握的一项技能。

本小节将对 VHDL 硬件描述语言的一些背景知识进行简单介绍,主要包括 VHDL 语言的历史、VHDL 语言的优缺点以及 VHDL 语言的设计流程。下面首先介绍 VHDL 硬件描述语言的历史。

### 1.2.1 VHDL 语言的历史

美国国防部在 20 世纪 70 年代末和 80 年代初提出了 VHSIC (Very High Speed Integrated Circuit) 计划,VHSIC 计划的目标是为下一代集成电路的生产、实现阶段性的工艺极限以及完成 10 万门级以上的设计建立一项新的描述方法。

1981 年,美国国防部提出了一种新的硬件描述语言 HDL,称为“超高速集成电路硬件描述语言”,简称 VHDL (VHSIC Hardware Description Language) 语言。当这个语言被首次开发出来时,其目标只是使电路文本化的一种标准,主要是为了使采用文本描述的设计能够为其他人所理解,同时也用作模型语言,能采用软件进行模拟。VHDL 语言的结构和设计方法受到了 ADA 语言的影响,并吸收了其他硬件描述语言的优点。1986 年,IEEE 致力于 VHDL 语言的标准化工作,为此成立了 VHDL 语言标准化小组。经过了多次的修改与扩充,直到 1987 年 12 月 VHDL 语言才被接纳为 IEEE 1076 标准。1988 年,Milstd454 规定所有为美国国防部设计的 ASIC 产品必须采用 VHDL 语言来进行描述。1993 年,IEEE 1076 标准被修订,更新为新的 VHDL 语言标准 IEEE 1164。1996 年,IEEE1076.3 成为 VHDL 语言的综合标准。

1995 年我国国家技术监督局制定的《CAD 通用技术规范》推荐 VHDL 语言作为我国电子设计自动化硬件描述语言的国家标准。至此,VHDL 语言在我国迅速普及,现在这门语言已经成为从事硬件电路设计的开发人员所必须掌握的一项技术。

### 1.2.2 VHDL 语言的主要优势

今天,VHDL 语言已经成为数字电路和系统的描述、建模和综合的工业标准。在电子产业界,无论是系统设计人员,还是 ASIC 设计人员或者各大中院校的学生都应该通过学习 VHDL 语言来提高他们的工作效率。由于 VHDL 语言的通用性,它现在已经成为支持不同层次设计者要求的一种标准硬件描述语言。VHDL 语言能够成为标准并且获得广

泛的应用，一定有它自身的主要优势，或者说是与众不同的特点。

### 1) 强大的功能和灵活性

VHDL 语言具有功能强大的语言结构，可以用简洁明确的程序来描述复杂的逻辑控制。为了有效控制设计的实现，它具有多层次的设计描述功能，支持设计库和可重复使用的元件生成；而且它还支持阶层设计和提供模块设计的创建。同时，VHDL 语言还支持同步电路、异步电路和随机电路的设计，这是其他的硬件描述语言所不能比拟的。

### 2) 独立于器件的设计

设计人员采用 VHDL 语言进行硬件电路的设计时，并不需要首先选择完成此项设计的逻辑器件。这样设计人员可以集中时间来进行硬件电路系统的具体设计，而不需要考虑其他的问题。当采用 VHDL 语言完成硬件电路系统的功能描述后，可以使用不同的逻辑器件来实现其功能。如果需要对设计进行资源利用和性能方面的优化，也并不要求设计人员非常熟悉器件的内部结构。这样，设计人员就可以集中精力来进行设计的构思。

### 3) 可进行程序移植

VHDL 语言的移植能力是允许设计人员对需要综合的设计描述进行模拟，在综合前对一个数千门的设计描述进行模拟可以节约大量可观的时间。由于 VHDL 语言是一种标准化的硬件描述语言，因此同一个设计的 VHDL 语言描述可以被不同的 EDA 工具支持，从而使得 VHDL 语言程序的移植成为可能。VHDL 语言的移植能力就是指同一个设计的 VHDL 语言描述可以从一个模拟工具移植到另一个模拟工具、从一个综合工具移植到另一个综合工具或者从一个工作平台移植到另一个工作平台。

### 4) 性能评估能力

独立于器件的设计和可进行程序移植允许设计人员可以采用不同的器件结构和综合工具来对自己的设计进行评估。在设计人员开始具体的设计之前，他们并不需要了解将采用何种逻辑器。设计人员可以进行一个完整的 VHDL 语言描述，并且可以对它进行综合，生成选定的器件结构的逻辑功能，然后再对设计结果进行评估，最后选用最适合该设计的逻辑器件。同样为了衡量综合的质量，设计人员可以采用不同的综合工具对设计进行综合，然后再对综合结果进行分析和评估。

### 5) 易于 ASIC 移植

VHDL 语言效率高的重要体现之一就是如果设计人员的设计是被综合到一个 CPLD 器件或 FPGA 器件，那么就可以使设计的产品以最快的速度上市。当产品的数量达到相当的规模时，采用 VHDL 语言能够很容易地帮助设计人员实现转成 ASIC 的设计。有时用于 PLD 的程序可以直接用于 ASIC；并且由于 VHDL 语言是一种 IEEE 的工业标准硬件描述语言，所以用 VHDL 语言设计可以确保 ASIC 厂商生产高质量的器件产品。

### 6) VHDL 语言标准、规范，易于共享和复用

VHDL 语言的语法规则、标准，可读性强。用 VHDL 语言书写的代码文件既是程序，又是文档；既是设计人员进行设计成果交流的交流文件，也可作为合同签约者之间的合同文本。另一方面，由于 VHDL 语言是一种 IEEE 的工业标准硬件描述语言，具有严格的语法规则和统一的标准，因此它可以使设计成果在设计人员之间进行交流和共享。反过来，就可以进一步推动 VHDL 语言的发展和完善。



### 1.2.3 VHDL 语言的不足之处

尽管 VHDL 语言作为 IEEE 的工业标准具有许多其他硬件描述语言所不具有的主要优势,同时它也存在一些不足之处。

#### 1) 需要了解较多的硬件电路知识

采用 VHDL 语言描述硬件电路需要设计人员具有较多的硬件电路知识,甚至包括目标芯片基本结构方面的知识。需要注意的是,设计人员在编写 VHDL 语言程序时,应该以一种并行思路去理解和应用 VHDL 语言。硬件电路系统的内部子系统乃至部分器件的工作状态和工作方式可以是相互独立、互不相关的,同时也可以前后相继、互为因果的。一般来讲,VHDL 语言对硬件电路的实际工作方式往往采用并行语句和顺序语句来描述在某一时刻可能会发生的事件。因此,设计人员在采用 VHDL 语言描述硬件电路时应该以一种并行的思路去进行硬件电路系统的描述。

#### 2) VHDL 语言的描述会与实际硬件电路的工作方式不符

设计人员采用 VHDL 语言描述硬件电路时,需要特别注意的是 VHDL 语言的描述会与实际硬件电路的工作方式不符。例如,多路选择器等以并行方式工作的组合电路在 VHDL 语言描述中却采用顺序语句描述。顺序语句一般是按照时钟的节拍来运行的,它是一种典型的计算机指令语句,而实际硬件电路的运行却并非如此。有关这方面的问题一定要引起读者们的注意。

#### 3) 不具有描述模拟电路的能力

VHDL 语言并不具有描述模拟电路的能力,研究结果证明 VHDL 语言虽然可以扩展到电路级上,但是在电路级上 VHDL 语言并不是一种理想的硬件描述语言。目前 IEEE 的 1076.1 小组正在设计一种新的硬件描述语言,能够描述模拟电路和数模混合电路。这个新的硬件描述语言将以 VHDL 语言为基础,并且在此基础上增加描述模拟电路的一些扩展内容。

### 1.2.4 VHDL 语言的设计流程

采用 VHDL 语言设计硬件电路系统的设计流程一般可以分为以下几个步骤。

- ① 硬件电路系统设计要求的定义。
- ② 编写描述硬件电路系统功能的 VHDL 语言程序。
- ③ VHDL 语言程序的模拟。
- ④ VHDL 语言程序的综合、优化和布局布线。
- ⑤ 布局布线后的设计模拟。
- ⑥ 器件的编程。

设计人员在从事硬件电路系统的设计过程中,编写 VHDL 语言程序之前必须对硬件电路系统的设计目的和设计要求有一个非常明确的认识才行。例如,硬件电路系统的设计功能是什么?硬件电路系统中的信号建立时间、信号保持时间和最大工作频率等技术指标的要求是什么?只有对硬件电路系统的设计目的和设计要求有了明确的认识以后,设计人员才可以选择适当的 VHDL 语言设计方式和实现功能的逻辑器件。